MPI

CONCOURS COMMUN INP

RAPPORT DE L'ÉPREUVE ORALE D'INFORMATIQUE

PROPOS INTRODUCTIFS

Le sujet de l'oral d'informatique CCINP en filière MPI est constitué de deux exercices :

- un exercice de type A, "théorique", noté sur 8 points et ne nécessitant pas l'utilisation d'un ordinateur. L'exercice A peut évaluer la syntaxe SQL mais ni la syntaxe C ni la syntaxe OCaml.
- un exercice de type B, "pratique", noté sur 12 points et consacré principalement à de la programmation en C ou en OCaml. Quelques questions d'analyse y sont également posées. Cet exercice fournit généralement un code compagnon que le candidat devra compléter, corriger ou modifier. Il est également possible que le candidat doive écrire intégralement le code demandé par l'énoncé.

L'oral, d'une durée totale d'une heure, se décompose en deux temps :

- 30 minutes de préparation du sujet sur feuille et machine, démarches administratives incluses.
- 30 minutes de passage avec examinateur, démarches administratives incluses.

Les rapports des sessions précédentes ainsi qu'une sélection de sujets de l'épreuve orale d'informatique sont publiés sous la rubrique ANNALES, SUJETS ET RAPPORTS DE LA FILIÈRE MPI :
https://www.concours-commun-inp.fr/fr/epreuves/annales/annales-mpi.html
Également, restent disponibles les 12 « exercices 0 » qui furent proposés à la rentrée 2022-23, en amont de la première session Oraux 2023 de la filière MPI, afin que les candidats se préparent au mieux à cette nouvelle épreuve.

Ce rapport rappelle les consignes principales, qui sont susceptibles d'évoluer. Se référer à la notice du Concours pour obtenir l'ensemble de l'information et au site du concours https://www.concours-commun-inp.fr/fr/epreuves/les-epreuves-orales.html rubrique MPI. Vous y trouverez le cadre de l'épreuve (dont l'environnement technique linux sous format fichier .ova).

En complément à ce rapport, une sélection de sujets, commentés et partiellement corrigés, posés lors de la session 2025 sera publiée, ainsi que l'archive correspondante contenant les codes compagnons, les codes corrigés et les fichiers sources LaTeX.

PRÉPARATION DU SUJET

La préparation s'effectue dans une salle dédiée. Plusieurs candidats préparent en même temps, dans des conditions favorisant la concentration, sous la surveillance de vacataires. À l'entrée de la salle de préparation, convocations et pièces d'identité sont vérifiées.

Pour préparer, le candidat dispose :

- d'un ordinateur disposant de l'environnement linux diffusé sur le site du concours à la rubrique MPI[1] et d'un clavier (configuration AZERTY par défaut, sans pavé numérique),
- de feuilles de brouillon,
- des énoncés imprimés des deux exercices (types A et B),
- d'une calculatrice,
- d'une clé USB contenant le code accompagnant l'exercice de type B.

Le candidat doit copier le code compagnon disponible sur la clé sur sa machine pour pouvoir le compléter / modifier, puis copier le code produit dans un répertoire dédié sur sa clé USB en fin de préparation. Nous insistons sur l'importance du respect de ces consignes qui permettent de disposer d'une sauvegarde du travail produit en préparation en cas de mauvaise manipulation de la clé USB. Les consignes quant aux modalités de préparation et notamment la manipulation des clés sont projetées dans la salle de préparation, de même qu'un chronomètre indiquant le temps de préparation restant. Ce dernier est fixé à 27 minutes, les 3 minutes restantes étant réservées aux formalités administratives et au déplacement vers les salles d'interrogation.

En fin de préparation, les candidats sont accompagnés jusqu'aux salles d'interrogation par un vacataire.

PASSAGE DU CANDIDAT

Le candidat rentre dans la salle qui lui est indiquée, fournit à l'examinateur sa feuille de passage et sa pièce d'identité puis signe la feuille d'émargement que lui présente l'examinateur. Le chronomètre est lancé à ce moment, pour un temps de passage effectif de 27 minutes.

Le candidat dispose d'un ordinateur prêt, dans la même configuration que celui qu'il a utilisé dans la salle de préparation. Un vidéoprojecteur permet de présenter son travail à l'examinateur. Le candidat est invité à copier le code présent sur sa clé sur la machine mise à disposition.

Le candidat commence par présenter tout ce qu'il a préparé. Il débute par l'exercice de son choix et peut, à tout moment, changer d'exercice et/ou revenir sur l'exercice précédemment abordé. Il dispose pour sa présentation d'un tableau et de son code source projeté. L'examinateur demandera que le code produit soit compilé / exécuté / interprété. Il peut donner des indications quant au temps restant et, le cas échéant, indiquer au candidat sur quelles questions se concentrer.

En fin d'épreuve, après la fin du temps imparti, le candidat remet à l'examinateur les énoncés des deux exercices, ses brouillons et la clé contenant le code produit. Il ferme, sur l'ordinateur, les fenêtres de son code et efface le tableau avant de récupérer sa pièce d'identité et sa feuille de passage signée par l'examinateur.

CRITÈRES D'ÉVALUATION

Sont prises en compte dans l'évaluation :

- la maîtrise des éléments du programme relatifs aux deux exercices,
- la maîtrise de l'environnement technique et des outils mis à disposition,
- la pertinence de la réflexion,
- l'interactivité lors de la présentation des résultats et des tests (exercice de type B),
- la réactivité aux éventuels conseils et indications de l'examinateur,
- la qualité de la prestation orale.

[1] https://www.concours-commun-inp.fr/fr/epreuves/les-epreuves-orales.html

BILAN

Sur 773 candidats admissibles (dont 120 grands admissibles), 574 se sont présentés aux épreuves orales. L'épreuve Informatique est conçue sur la base d'une banque commune d'exercices. La moyenne des notes obtenues est de 10,15 avec un écart-type de 3,63. Cela répond aux attendus du CCINP et démontre la capacité à classer les candidats.

Les examinateurs ont proposé des exercices couvrant l'ensemble du programme des deux années de MPI. En particulier, les thématiques suivantes ont été abordées : la récursivité, la représentation de types, les structures de données (listes, tableaux, piles, files, tas, etc.), les arbres, les graphes, les algorithmes d'approximation, les algorithmes probabilistes, les stratégies algorithmiques (diviser pour régner, programmation dynamique, retour sur trace, etc.), l'algorithmique des textes, la concurrence, la décidabilité, la NP-complétude, les langages réguliers, les automates finis, les grammaires non contextuelles, les algorithmes d'apprentissage, l'algorithmique des jeux, la déduction naturelle ou encore les bases de données.

Le langage des exercices de type B était imposé : environ 50 % des exercices devaient être traités en OCaml et 50 % en C. La syntaxe SQL a, quant à elle, été évaluée via certains exercices de type A.

Les examinateurs ont pu voir d'excellentes prestations et sont plutôt satisfaits du niveau global des candidats. La majorité des candidats maîtrise les langages OCaml et C de manière satisfaisante. Ils ont dans l'ensemble été bien préparés à cette épreuve aux modalités exigeantes. Cependant, un nombre non négligeable de candidats fait preuve d'une méconnaissance quasi totale des outils mis à disposition par l'environnement machine du concours. Cette méconnaissance a un effet délétère indéniable sur la prestation des candidats concernés et les handicape certainement encore plus en préparation. Le sentiment des examinateurs est que cet état de fait est lié à un manque d'entraînement pendant l'année avec des outils similaires. Ils soulignent l'importance d'exposer les candidats à ce genre d'environnement linux au moins pendant la préparation aux oraux.

CONSEILS AUX CANDIDATS

Nous souhaitons souligner les points suivants, à l'attention des futurs candidats.

Conseils quant à la gestion du temps

De nombreux candidats n'ont pas su gérer leur temps durant le passage, passant par exemple plus de 20 minutes sur l'exercice A et réfléchissant à des questions qu'ils n'ont pas abordées en préparation plutôt que de présenter ce qu'ils ont fait sur l'exercice B d'abord (ou inversement). Plus d'une fois, les examinateurs ont pu constater sur les brouillons des candidats que ces derniers avaient abordé en préparation des questions non présentées pendant l'oral. L'examinateur tente d'aider les candidats dans leur gestion du temps mais leur tâche est rendue difficile par le fait que les candidats n'annoncent que très rarement ce qu'ils ont traité et se lancent dans des questions non traitées avant de présenter ce qui a été fait en préparation. Nous déplorons également des candidats peu efficaces qui, une fois leur présentation achevée, papillonnent sans réellement se concentrer sur les questions qu'ils n'ont pas encore traitées, comme s'il était impossible de répondre aux questions qui n'ont pas été préparées.

Un sujet est composé de deux exercices et il n'est pas possible de faire de transfert de points entre les deux. Un candidat qui réussit brillamment l'exercice A et n'aborde pas l'exercice B ne pourra avoir une note supérieure à 8/20.

En conséquence, nous rappelons que :

- il est conseillé aux candidats de prendre connaissance des deux exercices : certains énoncés sont difficiles à traiter à l'oral sans y avoir un peu réfléchi en amont ;

- l'examinateur connaît le sujet, il n'est donc pas nécessaire pour le candidat de le réintroduire. Il est important de citer le numéro de la question à laquelle on répond mais inutile de lire son énoncé dans son intégralité ;

- il est recommandé aux candidats de commencer par annoncer, en début d'épreuve, les questions des deux exercices qui ont été traitées en préparation. Ceci permet à l'examinateur d'aider les candidats à gérer leur temps et de les interroger sur l'ensemble des questions abordées pendant la phase de préparation ;

- il n'est pas toujours judicieux de commencer par l'exercice A. De plus, il est possible de passer d'un exercice à l'autre pendant le passage, voire de ne pas traiter toutes les questions dans l'ordre (ou même d'en sauter quelques-unes) ;

- le candidat peut, bien entendu, modifier son code pendant l'oral pour corriger ses erreurs ou aborder des questions non traitées en préparation en les programmant devant l'examinateur sur la machine de la salle.

Conseils quant à la manipulation de la machine et de l'environnement

- Savoir éjecter proprement une clé fait partie des compétences attendues d'un élève de MPI. Nous rappelons donc qu'il ne faut pas arracher une clé USB sans précaution, en particulier alors que des documents qui y sont présents sont encore utilisés. Certains candidats n'ont pas respecté les consignes relatives à la manipulation des clés et des fichiers (indiquées devant la salle de préparation, dans la salle de préparation et à l'oral par les vacataires). Ces manquements rendent impossible la récupération de leur travail en cas de problème. Certains candidats ont dû ainsi reprendre l'exercice B depuis le début pendant le passage.

- Durant la session 2025, nous avons pu relever les propos suivants de la part de candidats : "J'ai du mal avec Linux, comment on revient sur la fenêtre du terminal ?", "Vous savez où est le terminal ? Je sais qu'il y a gcc mais on faisait ça avec copilot sur github.", "Je ne trouve pas le bouton pour compiler sous VSCode", "D'habitude on fait ça en ligne (après une demande d'évaluation de code OCaml)".

  Ces propos sont le symptôme d'un manque important de maîtrise de l'environnement machine linux utilisé au CCINP (et similaire aux environnements linux des autres concours). Ne jamais compiler de C via un terminal ou s'entraîner uniquement sur des plateformes en ligne ne permet pas de se préparer correctement à l'épreuve orale d'informatique du CCINP. Nous conseillons aux candidats de prendre en main l'environnement machine du CCINP (voir https://www.concours-commun-inp.fr/fr/epreuves/les-epreuves-orales.html, rubrique MPI) avant les épreuves orales.

- Il est attendu des candidats une maîtrise élémentaire d'un terminal de manière à pouvoir se déplacer dans une arborescence et lancer une commande de compilation de leur code. Sans être nécessaire, l'examinateur rappelle que, dans un terminal, la touche de tabulation permet souvent de compléter des morceaux de la commande en cours et que les flèches directionnelles permettent de naviguer dans l'historique des commandes. Connaître ces raccourcis est un gain de temps non négligeable pour les candidats qui tapent lentement.

- Si aucune connaissance concernant la commande `make` n'est attendue, tous les exercices B utilisant le langage C mettent à disposition des candidats un `Makefile` ainsi que des consignes complètes permettant de l'utiliser, sous réserve d'une familiarité minimale avec un terminal. Le `Makefile` est disponible ici :
  https://www.concours-commun-inp.fr/fr/epreuves/les-epreuves-orales.html (rubrique MPI) et les consignes sont en tête de tous les exercices publiés comportant du C.

  Lors du passage à l'oral, l'examinateur peut demander aux candidats de compiler leur code avec un *sanitizer* pour évaluer certains aspects de la gestion de la mémoire. Les commandes de compilation permettant de compiler avec ou sans *sanitizer* sont disponibles dans les consignes, mais il est souvent plus rapide d'utiliser `make` ou `make safe`.

- Il a été constaté chez certains candidats un manque de familiarité avec l'usage d'un clavier. Cela se manifeste par une vitesse de frappe très lente ou des difficultés à taper certains caractères (guillemets, chiffres, …). Cela n'est bien entendu pas pénalisé, mais ces candidats se sont retrouvés mécaniquement désavantagés. Il est à noter que les claviers des machines ne disposent pas de pavé numérique et sont par défaut en configuration AZERTY et que les candidats doivent y être habitués ou bien savoir en changer.

- Certains candidats ont demandé s'il était possible d'utiliser - par exemple - VSCode pour présenter leur code OCaml. Les examinateurs soulignent que les candidats sont libres d'utiliser les éditeurs de leur choix, dans la limite des outils disponibles dans l'environnement du CCINP.

- Selon l'éditeur choisi, le passage à la ligne automatique (word wrap) n'est pas activé par défaut. Cela empêche l'examinateur de lire l'intégralité du code. Il sera alors demandé au candidat d'activer cette option ou de passer à la ligne manuellement.

Conseils quant à la programmation

- Trop nombreux sont les candidats qui arrivent au passage sans avoir jamais évalué leur code en préparation. Cela se voit dès la première question de code. Ils passent alors un temps laborieux et inconfortable à corriger la question 1 pendant le passage et se retrouvent parfois à ne même pas pouvoir présenter le code des questions suivantes. L'épreuve est certes courte, mais coder trop rapidement aux dépens de l'évaluation et des tests n'est pas une bonne stratégie pour réussir l'exercice B. L'immense majorité des candidats qui ne compile / n'évalue / ne teste pas son code en préparation n'atteint pas la moyenne à cette épreuve.

- Les tests des fonctions demandées par l'exercice B font partie de l'évaluation et l'examinateur demande à voir le résultat de ces tests pendant le passage. Dans le langage OCaml, comme dans le langage C, il est possible et souhaitable d'écrire les tests à l'avance plutôt que de devoir les réécrire pendant le passage.

- Même si elle n'est pas explicitement au programme, la fonction Printf.printf en OCaml a un fonctionnement très similaire à celui de la fonction printf en C et est particulièrement pratique pour un oral, que ce soit comme outil simple de débogage ou pour valider simplement un programme par un affichage d'exemples. Le cas échéant, l'énoncé en rappelle les modalités d'usage et les codes de formatage.

- Les fonctionnalités des langages au programme ainsi que tout ce qui se trouve dans la rubrique "éléments de syntaxe devant être reconnus et utilisables après rappels" peuvent être librement utilisés par les candidats. Si un candidat utilise une construction des langages en dehors de ce cadre, il pourra lui être demandé d'expliquer comment répondre à la question avec uniquement les

outils du programme. L'examinateur n'interdit pas, mais ne souhaite en aucun cas encourager, l'utilisation d'éléments hors programme : tous les sujets peuvent être résolus avec le fragment au programme des langages et utiliser des constructions alternatives expose le candidat à des questions qui peuvent lui faire perdre du temps sans que sa réponse ne soit davantage valorisée.

- La manipulation élémentaire de la mémoire en C pose des problèmes importants à de nombreux candidats. L'usage élémentaire de `malloc` ou de `free` est trop souvent mal maîtrisé, en particulier en ce qui concerne la nécessité d'initialiser les variables contenant des pointeurs et la valeur du contenu de la mémoire allouée dynamiquement (non, il n'y pas des 0 par défaut lors de l'allocation dynamique d'un tableau).

- Nous conseillons vivement aux candidats de s'entraîner à lire et comprendre les messages d'erreurs dans les deux langages. En particulier pour le langage C, le *sanitizer* et l'option `-g` (dont l'usage est indiqué dans les commandes de compilation et implémenté dans le `Makefile`) permettent d'obtenir des messages d'erreur plus explicites et localisés qui aident à la correction de ces erreurs, pour peu qu'on sache les interpréter.

- On observe encore de trop nombreuses confusions entre listes et tableaux en OCaml avec des tentatives non pertinentes d'accéder à un élément d'une liste via son indice.

Conseils quant à la maîtrise du programme

- La syntaxe SQL est généralement maîtrisée mais de très nombreux candidats ont eu de grandes difficultés avec la modélisation de problèmes et le modèle entité-association. Même si aucun formalisme n'est spécifié par le programme, des diagrammes entités-association simples font partie des attendus que les candidats doivent savoir mettre en œuvre. Il en va de même pour les schémas relationnels et leur lien avec le modèle entité-association.

- La structure d'arbre binaire de recherche est mal connue et parfois confondue avec celle des tas binaires. La notion d'arbres binaires de recherche équilibrés a posé de nombreux problèmes aux candidats.

- Un algorithme aussi essentiel et central que le parcours de graphe n'est souvent pas maîtrisé. Les difficultés liées au manque de maîtrise de cet algorithme s'étendent à ses variantes, en particulier Dijkstra et A*.

- Les graphes de flot de contrôle sont au programme.

- L'existence d'un invariant est souvent confondue avec une preuve de correction : nous rappelons aux candidats qu'une fois l'invariant prouvé, il reste à l'exploiter, ce qui n'est pas toujours immédiat.

- Il est important de savoir ce qu'est un ordre bien fondé.

- Le chapitre IA du programme de MPI n'est souvent pas bien maîtrisé. Les candidats confondent les noms des algorithmes et peinent à expliquer leur principe.

Divers

Dans certaines salles de passage, le tableau sert aussi d'écran de projection. Il est attendu des candidats qu'ils n'écrivent pas sur la partie du tableau servant à la projection sans l'accord de l'examinateur, sous peine de rendre la projection, ainsi que ce qui est écrit, illisible.